What is Claimed:

1. A method for scheduling at least one resource of a coprocessor in a computing system having a processor and a coprocessor, comprising:

receiving by a kernel mode driver a command buffer from a user mode driver, wherein the command buffer is formulated based on a request for coprocessor resources;

generating by a kernel mode driver at least one direct memory access (DMA) buffer based on said command buffer and a resource list corresponding to at least one memory resource associated with processing said at least one DMA buffer by said coprocessor, wherein a set of at least one DMA buffer is maintained for each client context;

analyzing said resource list, and based on said analyzing, generating a paging DMA buffer including at least one instruction that instructs the coprocessor with respect to the at least one memory resource; and

submitting said at least one DMA buffer to said coprocessor for processing.

2. A method according to claim 1, wherein the set of at least one DMA buffer is a software queue maintained for each application submitting requests for coprocessor resources.

3. A method according to claim 1, further comprising transmitting said at least one instruction to the coprocessor.

4. A method according to claim 1, wherein said submitting includes submitting said paging buffer to the coprocessor.

5. A method according to claim 1, wherein said generating includes assigning priority to elements in the at least one DMA buffer.

6. A method according to claim 1, wherein said paging buffer includes at least one command that requests the coprocessor to stop processing.

7. A method according to claim 6, wherein said at least one command is a fence that generates an interrupt and stalls the GPU.

8. A method according to claim 1, further comprising patching at least one reference to the at least one memory resource with actual memory resource data.

9. A method according to claim 1, whereby said coprocessor processes said at least one DMA buffer in a different order than the order in which said at least one DMA buffer is generated.

10. A method according to claim 1, wherein said coprocessor is busy processing another DMA buffer during at least one of said receiving, generating, analyzing and submitting.

11. A method according to claim 1, further comprising:

interrupting the processing of a DMA buffer submitted to the coprocessor according to said submitting; and

resuming processing of the interrupted DMA buffer.

12. A method according to claim 1, wherein said analyzing includes determining whether said at least one DMA buffer implicates a threshold amount of memory resources.

13. A method according to claim 12, wherein if said at least one DMA buffer implicates more than the threshold amount, splitting said at least one DMA buffer.

14. A method according to claim 1, wherein said at least one instruction includes at least one of an evict, page in and relocate instruction.

15. An application programming interface comprising computer executable modules having computer executable instructions for carrying out the method of claim 1.

16. A computing device comprising means for carrying out the method of claim 1.

17. A modulated data signal carrying computer executable instructions for performing the method of claim 1.

18. A computing device having a scheduling component for scheduling at least one resource of a coprocessor in a computing system having a processor and a coprocessor, comprising:

an input interface for receiving by the scheduling component a command buffer from a user mode driver, wherein the command buffer is formulated based on a request for coprocessor resources from a particular client context;

a translator component that generates at least one direct memory access (DMA) buffer based on said command buffer and a resource list corresponding to at least one memory resource associated with processing said at least one DMA buffer by said coprocessor, wherein a set of at

least one DMA buffer is maintained for each client context;

a memory component for analyzing said resource list, and based on said analyzing, generating a paging DMA buffer including at least one instruction that instructs the coprocessor with respect to the at least one memory resource; and

an output interface for submitting said at least one DMA buffer to said coprocessor for processing.

19. A computing device according to claim 18, wherein the set of at least one DMA buffer is a software queue maintained for each application submitting requests for coprocessor resources.

20. A computing device according to claim 18, wherein said at least one instruction is transmitted to the coprocessor via the output interface.

21. A computing device according to claim 18, wherein said paging buffer is submitted to the coprocessor via the output interface in conjunction with submitting said at least one DMA buffer.

22. A computing device according to claim 18, wherein priority is assigned to elements in the at least one DMA buffer.

23. A computing device according to claim 18, wherein said paging buffer includes at least one command that requests the coprocessor to stop processing.

24. A computing device according to claim 23, wherein said at least one command is a fence that generates an interrupt and stalls the GPU.

25. A computing device according to claim 18, wherein at least one reference to the at least one memory resource is patched with actual memory resource data prior to submission of the at least one DMA buffer.

26. A computing device according to claim 18, whereby said coprocessor processes said at least one DMA buffer in a different order than the order in which said at least one DMA buffer is generated.

27. A computing device according to claim 18, wherein said coprocessor is busy processing another DMA buffer while said at least one DMA buffer is being generated.

28. A computing device according to claim 18, wherein processing of a DMA buffer submitted to the coprocessor is interrupted and then processing of the interrupted DMA buffer is resumed.

29. A computing device according to claim 18, wherein said memory component determines whether said at least one DMA buffer implicates a threshold amount of memory resources.

30. A computing device according to claim 29, wherein if said at least one DMA buffer implicates more than the threshold amount, said at least one DMA buffer is split.

31. A computing device according to claim 18, wherein said at least one instruction includes at least one of an evict, page in and relocate instruction.

32. A method for scheduling at least one resource of a coprocessor in a computing system having a processor and a coprocessor, comprising:

means for receiving by a kernel mode driver a command buffer from a user mode driver, wherein the command buffer is formulated based on a request for coprocessor resources;

first means for generating by the kernel mode driver at least one DMA buffer based on said command buffer and a resource list corresponding to at least one memory resource associated with processing said at least one direct memory access (DMA) buffer by said coprocessor, wherein a set of at least one DMA buffer is maintained for each client context;

means for analyzing said resource list,

second means for generating a paging DMA buffer, based on said analyzing by said means for analyzing, including at least one instruction that instructs the coprocessor with respect to the at least one memory resource; and

means for submitting said at least one DMA buffer to said coprocessor for processing.

33. A computing device according to claim 32, wherein the set of at least one DMA buffer is a software queue maintained for each application submitting requests for coprocessor resources.

34. A computing device according to claim 32, further comprising means for transmitting said at least one instruction to the coprocessor.

35. A computing device according to claim 32, wherein said means for submitting includes means for submitting said paging buffer to the coprocessor.

36. A computing device according to claim 32, wherein said first means for generating includes assigning priority to elements in the at least one DMA buffer.

37. A computing device according to claim 32, wherein said paging buffer includes at least one command that requests the coprocessor to stop processing.

38. A computing device according to claim 37, wherein said at least one command is a fence that generates an interrupt and stalls the GPU.

39. A computing device according to claim 32, further comprising means for patching at least one reference to the at least one memory resource with actual memory resource data.

40. A computing device according to claim 32, whereby said coprocessor processes said at least one DMA buffer in a different order than the order in which said at least one DMA buffer is generated by said first means for generating.

41. A computing device according to claim 32, wherein said coprocessor is busy processing another DMA buffer during operation of at least one of said means for receiving, first and second means for generating, means for analyzing and means for submitting.

42. A computing device according to claim 32, further comprising:

means for interrupting the processing of a DMA buffer submitted to the coprocessor according to said submitting; and

means for resuming processing of the interrupted DMA buffer.

43. A computing device according to claim 32, wherein said means for analyzing includes means for determining whether said at least one DMA buffer implicates a threshold amount of memory resources.

44. A computing device according to claim 43, further comprising means for splitting a DMA buffer, wherein if said at least one DMA buffer implicates more than the threshold amount, said means for splitting splits said at least one DMA buffer.

45. A computing device according to claim 32, wherein said at least one instruction includes at least one of an evict, page in and relocate instruction.